Prerequisites:

1. A BeagleBone Black with latest debian running on it
2. Toolchain downloaded from Linaro website (Since Beaglebone's latest debian version uses GCC 6.3 I'm using 6.3 from linaro. Make sure to check GCC version from linaro to the one on your Beaglebone.)
3. Basic cmd knowledge

Procedure:

1. **[HOST]** Download the latest qt archive from this link. You can change the version of Qt by browsing Qt archive site as shown below:
   `qt >> "Select latest qt version at top" >> "Select latest subversion (i.e. this tuto uses 5.13.1)" >> single >> Download latest qt-everywhere.tar.xz`

2. **[HOST]** Make a directory called **BBB** in your home directory. (`mkdir ~/BBB`) and `cd` into this directory.

3. **[HOST]** Extract the Toolchain in this folder.

   ```
   unxz gcc-linaro-6.3.1-2017.05-x86_64_arm-linux-gnueabihf.tar.xz
   tar -xvf gcc-linaro-6.3.1-2017.05-x86_64_arm-linux-gnueabihf.tar
   ```

4. **[HOST]** Make a folder for sysroot in this directory. (`mkdir sysroot`)
   (Significance of sysroot)

5. **[HOST]** To sync the sysroot from beaglebone to your computer, use following commands:

   ```
   rsync -avz debian@beaglebone.local:/lib sysroot
   rsync -avz debian@beaglebone.local:/sbin sysroot
   rsync -avz debian@beaglebone.local:/usr sysroot
   ```

   The relative address `beaglebone.local` stands for ip address of your beaglebone black. If its connected over other network, then use that address.

6. **[HOST]** Check the directories in sysroot folder if they're populated with files from beaglebone.

7. **[HOST]** Make sure you're in the **BBB** directory and download sysroot-relativelinks file from github.

8. **[HOST]** Add executable flag to the file you created/downloaded above with the command:
   `sudo chmod a+x sysroot-relativelinks.py`
   Run the python script as follows:
   `./sysroot-relativelinks.py sysroot`
   (Its necessary to specify the foldername after the command.)

9. **[HOST]** Extract the downloaded Qt archive in suitable directory. For the sake of tutorial, I'm extracting it in `home` directory.

   ```
   unxz qt-everywhere-src-x.xx.x.tar.xz
   tar -xvf qt-everywhere-src-x.xx.x.tar
   mv qt-everywhere-src-x.xx.x ~/qt-5.xx
   ```

   `cd` into the newly created qt-5.xx directory and proceed to configure Qt for beaglebone.

10. **[HOST]** Run following command in Qt directory:

    ```
    ./configure -platform linux-g++ -release -device linux-beagleboard-g++
    -sysroot /home/"username"/BBB/sysroot -prefix /home/debian/Qt5ForBBB -
    hostprefix ~/Qt5forBBB -device-option
    CROSS_COMPILE=/home/"username"/BBB/gcc-linaro-6.3.1/bin/arm-linux-
    gnueabihf- -nomake tests -nomake examples -no-opengl -opensource -
    confirm-license -reduce-exports -make libs -no-egl -no-eglfs -qt-zlib
    -qt-libpng -no-use-gold-linker -linuxfb -qt-libjpeg -no-openssl -no-
    cups -no-glib -no-iconv -nomake examples -nomake tools -nomake tests -
    no-pkg-config -skip qtdeclarative -skip qtlocation -skip qtpurchasing
    -skip qtwayland -skip qtwebchannel -skip qtwebengine -skip
    qtwebsockets -skip qtwebview -skip qtwebglplugin -skip qtandroidextras
    -skip qtgamepad -skip qtmacextras -skip qtwinextras -skip qtsensors -
    skip qtconnectivity -recheck-all -v
    ```

    Verify all the paths in the command before pasting it into the terminal. You can follow the output if its successful or not. Common errors generally involve wrong paths to toolchain or sysroot.

11. **[HOST]** Run `make -j4` to generate makefiles for each module specified. This process will take atleast 1 hour to complete. If all goes well you'll have compiled Qt5.x for your beaglebone successfully.

12. **[HOST]** Run `make install` to finalise the installation process. After successful installation, you'll have a directory called Qt5For BBB in following paths: `~/Qt5ForBBB` and `~/BBB/sysroot/home/debian/Qt5ForBBB`.

13. **[HOST]** The directory in `~/BBB/sysroot/home/debian/Qt5ForBBB` contains all the compiled libraries and plugins necessary to run Qt on Beaglebone black. It needs some fonts to display the GUI on BBB, so download some free fonts. I use DejaVu fonts as they're free. Make sure you've fonts in .ttf and copy those .ttf files to `Qt5ForBBB/lib/fonts` directory. To copy the Qt5ForBBB directory to beaglebone use the following command:

    ```
    scp -r Qt5ForBBB debian@beaglebone.local:Qt5ForBBB
    ```

    (This will copy the folder from your computer to beaglebone's home directory.)

14. **[TARGET BBB]**
    So, on the beaglebone we need to setup the environment variables to ensure that Qt can find its compiled libraries in **Qt5ForBBB** folder. To do this, enter these commands one by one.

    ```
    export QT_QPA_FONTDIR=/home/debian/Qt5ForBBB/lib/fonts
    export
    QT_QPA_PLATFORM_PLUGIN_PATH=/home/debian/Qt5ForBBB/plugins/platforms
    export LD_LIBRARY_PATH="/home/debian/Qt5ForBBB/lib":$LD_LIBRARY_PATH
    export QT_PLUGIN_PATH=/home/debian/Qt5ForBBB/plugins
    ```

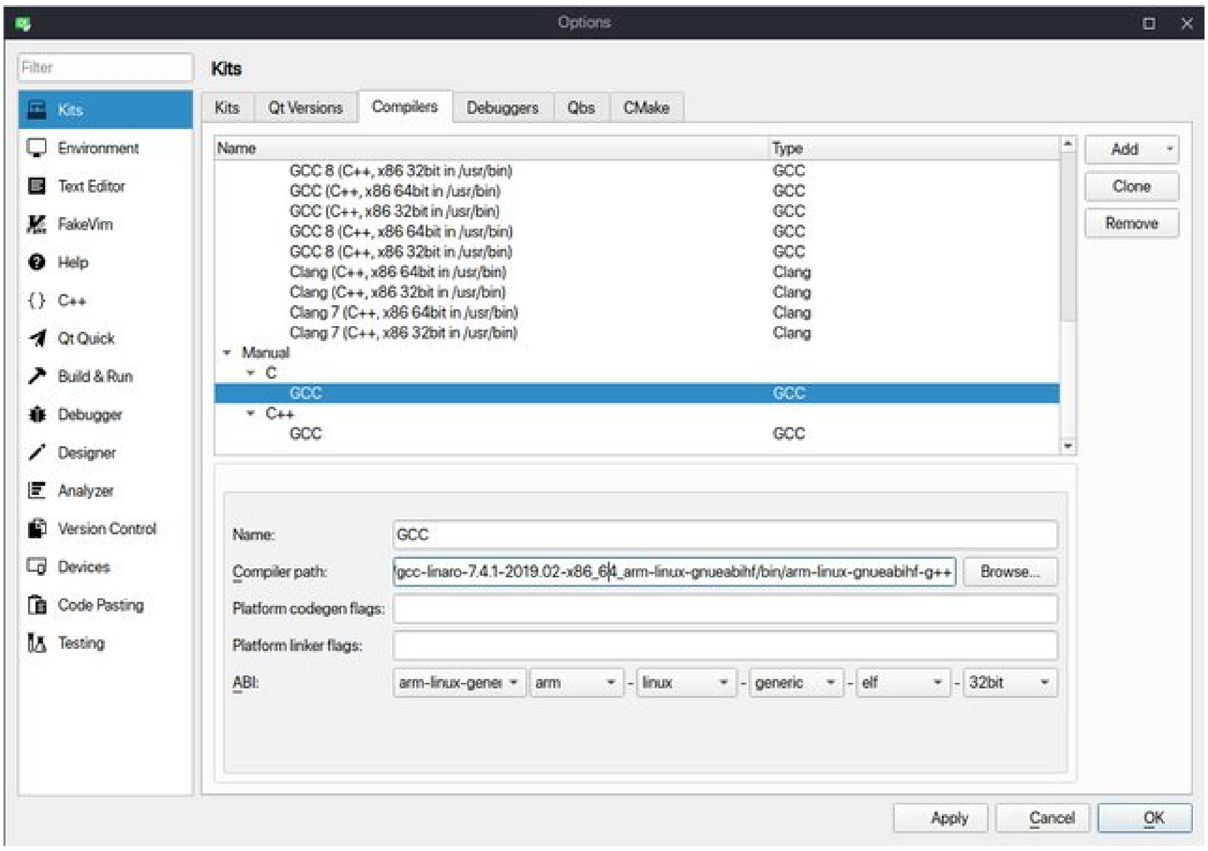
    (Use follwing command if you want to run the programs using VNC. You can use linuxfb or eglfs as well depending on your needs. Check available platform plugins in Qt5ForBBB/plugins/platforms)

    ```
    export QT_QPA_PLATFORM=vnc
    ```

    You can automize these commands by adding these 5 lines in your `~/.bashrc` file or you can make custom shell script with these 5 lines and put that script in `/etc/profile.d/` directory.

15. **[HOST] (Steps below are for Qt creator configuration)**

    - Add the `g++` executable from your linaro toolchain's folder as custom C and C++ compilers. Make sure select the same g++ executable as both or the Qt Creator application will give you an error. It should look something like this:



Name	Type
GCC 8 (C++, x86 32bit in /usr/bin)	GCC
GCC (C++, x86 64bit in /usr/bin)	GCC
GCC (C++, x86 32bit in /usr/bin)	GCC
GCC 8 (C++, x86 64bit in /usr/bin)	GCC
GCC 8 (C++, x86 32bit in /usr/bin)	GCC
Clang (C++, x86 64bit in /usr/bin)	Clang
Clang (C++, x86 32bit in /usr/bin)	Clang
Clang 7 (C++, x86 64bit in /usr/bin)	Clang
Clang 7 (C++, x86 32bit in /usr/bin)	Clang
Manual	
C	
GCC	GCC
C++	
GCC	GCC

Name: GCC
Compiler path: gcc-linaro-7.4.1-2019.02-x86_64_arm-linux-gnueabihf/bin/arm-linux-gnueabihf-g++
Platform codegen flags:
Platform linker flags:
ABI: arm-linux-generic - arm - linux - generic - elf - 32bit

- Set Qt version in Qt creator: Go to `Tools > Options > Kits > Qt versions > Add` and browse the qmake in `/bin` folder in Qt installation directory specified with -prefix parameter above in configuration step.
  (For example: in my case it was in `~/Qt5forBBB/bin/qmake`)

- Final step is to add BeagleBone as Kit in Qt Creator:
  To combine all the settings above, go to `Tools > Options > Kits` and then click Add.
  Configure as follows:

  ```
  Name:                     BeagleBone (or as you wish.)
  Device type:              Generic Linux Device
  Device:                   BBB (Or as you have named in Devices tab in
  Options)

  Compiler C & C++:         As configured in step above
  Debugger:                 Use gdb-multiarch
  (gdb-multiarch is available on Ubuntu, arch and other popular
  linux distros.)

  Qt version:               Same one from step above.
  ```

  Then click Apply and Ok.